

It's a Tough Nanoworld: in Tile Assembly, Cooperation is not (strictly) more Powerful than Competition

Florent Becker*

Pierre-Étienne Meunier[†]

Abstract

We present a strict separation between the class of “mismatch free” self-assembly systems and general aTAM systems. Mismatch free systems are those systems in which concurrently grown parts must always agree with each other.

Tile self-assembly is a model of the formation of crystal growth, in which a large number of particles concurrently and selectively stick to each other, forming complex shapes and structures. It is useful in nanotechnologies, and more generally in the understanding of these processes, ubiquitous in natural systems.

The other property of the local assembly process known to change the power of the model is cooperation between two tiles to attach another. We show that disagreement (*mismatches*) and cooperation are incomparable: neither can be used to simulate the other one.

The fact that mismatches are a hard property is especially surprising, since no known, explicit construction of a computational device in tile assembly uses mismatches, except for the very recent construction (FOCS 2012) of an intrinsically universal tileset, i.e. a tileset capable of simulating any other tileset up to rescaling. This work shows how to use intrinsic universality in a systematic way to highlight the essence of different *features* of a model.

Moreover, even the most recent experimental realizations do not use competition, which, in view of our results, suggests that a large part of the natural phenomena behind DNA self-assembly remains to be understood experimentally.

1 Introduction

The formidable biological diversity of living organisms is due for a large part to the amazing richness of the computational processes at the core of natural processes, at the molecular level. A largely studied molecular interaction paradigm

*Univ. d'Orléans, INSA Centre Val de Loire, LIFO EA 4022, FR-45067 Orléans, France, florent.becker@univ-orleans.fr. Supported in part by ANR Grant / Avec le soutien du programme ANR “QuasiCool”.

[†]Department of Computer Science, Aalto University, Helsinki, Finland, pierre-etienne.meunier@aalto.fi. This research was supported in part by National Science Foundation Grant CCF-1219274.

is *cooperation* between molecules, in fields such as proteomics [3]. However, in tile self-assembly – a computing model inspired by such processes – *competition* seems to be at least as important as cooperation. In this paper, we investigate the relations between these two interaction paradigms, and conclude that both are necessary for self-assembly to retain its full computational power.

Tile self-assembly is a model of molecular growth introduced by Winfree [35] to study the molecular engineering possibilities offered by the components designed by Seeman [30] using DNA. This model studies the autonomous assembly of independent atomic components, in an unsupervised way. Using these ideas and their later developments, researchers have been able to build a number of structures, ranging from regular arrays [37] to fractal structures [14, 28], smiling faces [26, 34], DNA tweezers [39], logic circuits [23, 29], neural networks [24], or molecular robots [18]. These examples not only demonstrate our ability to craft nano-things in bottom-up processes, as opposed to traditional top-down crafting processes; they also provide a strong link between the theoretical study of tile self-assembly, and actual processes occurring in nature, as observed in wet-lab experiments.

In tile self-assembly, we consider the assembly of non-flippable, non-rotatable square tiles, with a *glue color* and an integer *glue strength* on each side, to an existing assembly. We consider a finite amount of *tile types*, with an infinite supply of each type. The dynamics starts from a *seed* assembly and proceeds asynchronously and non-deterministically, one tile at a time. At each step, a tile can stick to an existing assembly if the sum of glue strengths with neighboring tiles matching its glue colors is at least a parameter of the model called the *temperature*. This means that there may be mismatches between adjacent tiles; the only requirement is that there be enough glue strength for each tile to attach to the assembly.

This model is similar to Wang tilings [2, 25, 32, 33], essentially augmented with a mechanism for sequential, asynchronous growth, and in which mismatches are allowed. Despite its simplicity, it is very expressive, as it can simulate the behavior of any Turing machine [27], and produce arbitrary connected shapes with a number of tile types within a logarithmic factor of their size [31, 36].

Intrinsic simulations Our purpose in this work is to compare the dynamics of different models of assembly. A first objection to this, is that adding new tile types can change dramatically the computational power of our model: therefore, our comparisons would be not be very useful in the qualitative study of molecular processes if they consisted only of statements of the form “*model X can do something with one billion tile types, that model Y cannot do with ten tile types*”.

Getting round this objection is the purpose of *intrinsic simulations*, a notion of dynamic simulation *up to rescaling*. This idea originated in cellular automata, and has given rise to a large literature [1, 4, 6, 7, 21], and has also been studied in Wang tiling [15–17]. More recently, it has been adapted to tile assembly [9–11, 20, 38], the main difficulties of this adaptation being that tile assembly is *asynchronous*, and that the geometry of assemblies plays an important role, since in most models, a tile that has grown cannot detach from the assembly nor change its type, contrarily to the case of cellular automata.

Intrinsic simulations shall not be seen as a way to bypass the difficulty of proving negative results about tile self-assembly: indeed, a major achievement of

this line of research is the existence of an intrinsically universal tileset [10], i.e. a tileset that can simulate the behavior of any other modulo rescaling. This notion can therefore be used to separate different models or classes, by proving – as done in [20], and as we do in the present work – that a model cannot simulate arbitrary tile assembly systems, whereas arbitrary tile assembly systems can. Intrinsic universality has also allowed to identify important challenges such as the existence of an intrinsically universal tileset with a single (rotatable) polygonal tile [8], giving new insights on the long-standing open problem of the existence of an aperiodic tiling of the plane, with a single (rotatable) tile.

Moreover, it has been shown that intrinsic universality is distinct from Turing universality: for instance, the three-dimensional variant of non-cooperative tile assembly can simulate Turing machines [5], but cannot intrinsically simulate arbitrary tile assembly systems [20].

Mismatch-free and locally consistent systems An important class of tile assembly system, that has been extensively studied in the context of *non-cooperative* (i.e. temperature 1) tile assembly, is the class of systems whose productions never have mismatches between adjacent tiles. It has been shown [12, 19] that these systems are only capable of assembling semi-periodic shapes. However, a long-standing open problem of tile assembly is the decidability of this condition for non-cooperative systems. Moreover, in the three-dimensional generalization of tile assembly, mismatches are an essential aspect of the computational capabilities of the non-cooperative model [5].

At temperature at least 2, the situation is quite different, since almost all known constructions, with the exception of the intrinsically universal tilesets that have been built [8, 10], never make mismatches between adjacent tiles. Moreover, an intermediate result on the road to intrinsic universality was proven for the restricted class of *locally consistent* tile assembly systems [11], that are mismatch-free systems in which it is additionally required that all tiles attach with the sum of glue strengths *exactly* equal to the temperature, whereas in the regular model, tiles can also attach with strengths summing up to more than the temperature.

Moreover, as stated earlier in the introduction, most of the behaviors observed in natural systems are studied with *cooperation* in mind; the question is therefore asked, of whether these assumptions (the mismatch-free property, and local consistency) really weaken the model. More precisely, the construction in [10] of an intrinsically universal tileset relies crucially on mismatches: Sections 5.2.1, 5.3.1, 5.4.1 are typical examples where competition seems unavoidable.

1.1 Main results

Our results show the “feature-optimality” of these constructions, by proving that cooperation and redundancy *are actually* necessary to retain the full power of the model. They also show the relevance of intrinsic universality, as opposed to only computational universality, to study natural systems: indeed, in tile assembly, computational universality can be achieved using a very weak subset of the model’s features, and therefore fails to capture key elements of processes occurring in nature.

Finally, our results call for a deeper discussion with experimentalists: indeed, the most recent experimental implementations of computational processes using

tiles [13] do not use competition nor glue redundancy. If these are really happening in nature, our work gives examples of phenomena allowed by tile assembly, that ought to be demonstrated experimentally to achieve a full understanding of crystal growth.

More precisely, the features we study are:

- cooperation, through increases of the *temperature*;
- competition, through the presence of mismatches;
- redundancy, through attachment strength exceeding the temperature;
- and geometry, through the dimension of the space the assembly takes place in.

The two questions we aim to answer are, for each class, does it have a *complete* system, i.e. a system that is intrinsically universal for systems that are in the class itself.

The following is known:

- non-planarity can sometimes replace cooperation: temperature 1 systems in three dimensions can simulate planar temperature 2 zig-zag systems, which can simulate arbitrary Turing machines [5];
- cooperation is a matter of getting the temperature to 2, more is superfluous: there is an intrinsically universal tileset at temperature 2, for any dimension, when mismatches and over-attachments are allowed [10];
- cooperation cannot be simulated by temperature 1 systems [20];
- *locally consistent systems*, the ones that exhibit neither competition nor redundancy have a complete tileset, which works at temperature 2.

We complete this study by showing the following:

Theorem 1.1. *Competition cannot be simulated by cooperation: there is a temperature 1 system with mismatches which cannot be simulated by any system without mismatches. This remains true whether or not redundancy is allowed.*

Theorem 1.2. *Redundancy cannot be simulated either: there is a temperature 1 system with attachments exceeding the temperature which cannot be simulated at any temperature by locally consistent systems, i.e. without redundancy or competition.*

Theorem 1.3. *There is an intrinsically universal tileset for the class of systems without mismatches in \mathbb{Z}^3 (the question remains open in \mathbb{Z}^2).*

The complexity landscape of self-assembly we get is described on Figure 1.

Known constructions such as found in [10] use competition a lot to ensure that choices are unique. Theorem 1.1 shows that this is in fact necessary, at least if the simulated system has mismatches. In other words, cooperation and competition are actually different phenomena which cannot simulate each other.

Theorem 1.3 shows that this tension between competition and cooperation is weaker in three-dimensional systems, as cooperation can be simulated without competition. This parallels the result of [22] showing that in three dimensions, some aspects of cooperation such as Turing-completeness can be simulated using competition.

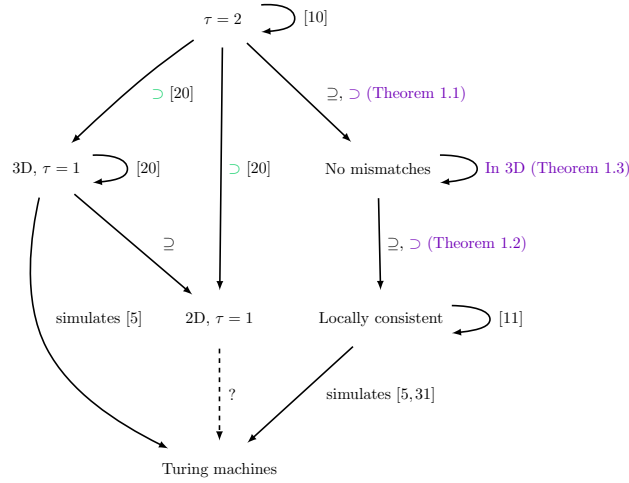


Figure 1: The landscape of behavior complexity in self-assembly. The comparisons in purple are the main results of this paper. Models with a loop have a complete tileset, i.e. one that is intrinsically universal for the class itself. The dashed arrow represents a conjectured comparison. Very few separation results between classes were previously known; on this diagram, they are written in green.

1.2 Key ideas

The main technical contributions of the paper are Lemma 3.4 (called the “*bisimilarity lemma*”), and the cutspace idea found in theorems 1.1 and 1.2. These are two powerful new tools for establishing negative results in self-assembly. There is a wealth of positive results (constructions) in self-assembly literature, but negative results are rare in comparison. One of the reasons for this is the difficulty of analyzing the dynamics of self-assembly systems, mainly because the intuitions from usual one-dimensional computing models (such as automata and Turing machines) are often false when more complex geometries are involved (even \mathbb{Z}^2).

The bisimilarity lemma (Lemma 3.4) is a very powerful tool for exploring the limits of tile assembly dynamics. Fundamentally, it is a “pumping lemma” for tile assembly systems. When manipulating such systems, it comes quite naturally to see that long and narrow part of the assembly act as wires with limited capacity. A few proofs have relied on that characteristic to prove that some part of an assembly can not possibly get enough information to behave as wanted. Yet, these proofs were tied to the geometry of such assemblies and what information they bore. The bisimilarity lemma gives a more comprehensive account of these limitations: on such a wire, communication might be bidirectional and tile additions may bring information in either direction. On a “wire” of finite width, only a small number of “communications” can be performed between distinct parts of the assembly.

The heavy lifting for this lemma lies not in its proof, but in the distinction between the *diplomatic set* representing what can be seen of the assembly

through some cut-set of the grid and the *policy set* of what actually happens there. A simple analogy for this lemma is: in order to know what happens in the world outside Las Vegas, you only need to know when and in what state people going into Clark County will come back out. What actually happens in there is of no consequence to the outside world¹.

This result is a generalization of a previous result called the *window movie lemma* [20]: indeed, the window movie lemma allows to “pause” the dynamics of self-assembly at some step, “move” parts of the assembly around, and then resume the dynamics. The *bisimilarity lemma* allows to repeat this operation: we can pause the dynamics, move parts by some vector \vec{v} , resume the dynamics for some time, and then move the parts back in place, with the current production. This operation can be repeated many times *with the same translation vector* \vec{v} .

Mismatchless systems and the cutspace. Once we have the right tool to bound the amount of information that is exchanged between different parts of the assembly, we also need a qualitative limit on what can be done without mismatches. Experimenting with such systems yields a first insight: without mismatches, concurrent assemblies “pass through each other”, and each of them will act as if it were the first one to come at a given spot. This is most acute whenever we have the archetypical race condition: two lines growing concurrently towards some point, the first come blocking the second from passing through.

To turn this intuition into a formal tool, we introduce the idea of *cutspace*s. The idea is to consider a tile assembly system, and to change the graph on which the assembly takes place. By doing so, we can observe what happens when all concurrent sub-assemblies are “*simultaneously* the first” to come at each race position. The key idea is to design a graph where each part of the assembly “sees” a different copy of the race position. Thus, they really all come first, in their own “parallel universe” where they have won the race. We then need to design a strategy to collapse these universes back into the same real spot and observe that without mismatches and without backdoor communication (ruled out through the bisimilarity lemma), they must all behave as they did on their own. Finally, all of them proceed at once as if they had won, contradicting the hypothesis that they are simulating a mismatch.

Three dimensional constructions. Theorem 1.3, an intrinsically universal tileset for mismatchless systems, relies on the fact that in non-planar grids, it is possible to wire up simulations without much limitation. In planar simulations of self assembly, it is crucial that either the set of inputs can be defined unambiguously, which rules out competition and redundancy in the simulated system, or that a race mechanism allows one of the possible sets of inputs to be selected as the winner, which requires competition in the simulator. In Theorem 1.3, we get rid of this constraint by simulating (up to rescaling) all possible sets of input sides for a tile simultaneously. Since we are simulating systems without mismatches, the representations of the possible outputs will always agree with neighboring tiles, even if these come afterwards.

¹More to the point, Vegas does not need an actual “real world” around it, it just needs cable TV in order to know how the world reacts to what it sees of Vegas.

2 Definitions and preliminaries

2.1 Tile self-assembly

We begin by defining the two-dimensional (respectively three-dimensional) abstract tile assembly model. A *tile type* is a unit square or hexagon (respectively cube) with four or six (respectively six) sides, each consisting of a glue *label* and a nonnegative integer *strength*. In the square grid of the plane, we call a tile's sides north, east, south, and west, respectively, according to the following picture:

We assume a finite set T of tile types, but an infinite supply of copies of each type. An *assembly* is a positioning of the tiles on a regular graph $G = (V, E)$ of degree four called the *space graph*, that is, a partial function $\alpha : V \dashrightarrow T$.

In the case of the square grid in two dimensions, and of the cubic grid in the plane, G is the Cayley graph of a commutative group, with generators $(1, 0)$ and $(0, 1)$ in the square grid of \mathbb{Z}^2 , and $(1, 0, 0)$, $(0, 1, 0)$ and $(0, 0, 1)$ in the cubic grid of \mathbb{Z}^3 . Therefore, in the rest of this paper, when there is no ambiguity, we will use the elements of the groups both as vectors and as points of the space. We will however use other kinds of graphs to formalize the idea of *cutspace*.

We say that two tiles in an assembly *interact*, or are *stably attached*, if the glue labels on their abutting side are equal, and have positive strength. An assembly α induces a weighted *binding graph* $B_\alpha = (V_\alpha, E_\alpha)$, which is a subgraph of G , where $V_\alpha = \text{dom } \alpha$, and there is an edge $(a, b) \in E_\alpha$ if and only if a and b interact, and this edge is weighted by the glue strength of that interaction. The assembly is said to be τ -stable if any cut of G has weight at least τ .

A *tile assembly system* is a triple $\mathcal{T} = (T, \sigma, \tau)$, where T is a finite tile set, σ is called the *seed*, and τ is the *temperature*.

Given two τ -stable assemblies α and β , we say that α is a *subassembly* of β , and write $\alpha \sqsubseteq \beta$, if $\text{dom } \alpha \subseteq \text{dom } \beta$ and for all $p \in \text{dom } \alpha$, $\alpha(p) = \beta(p)$. We also write $\alpha \rightarrow_1^\tau \beta$ if we can get β from α by the binding of a single tile, that is, if $\alpha \sqsubseteq \beta$ and $|\text{dom } \beta \setminus \text{dom } \alpha| = 1$. We note such a binding $t@z$ (read “tile t at position z ”), where $\{z\} = \text{dom } \beta \setminus \text{dom } \alpha$, and $t = \beta(z)$. We say that γ is *producible* from α , and write $\alpha \rightarrow^\tau \gamma$ if there is a (possibly empty) sequence $\alpha = \alpha_1, \dots, \alpha_n = \gamma$ such that $\alpha_1 \rightarrow_1^\tau \dots \rightarrow_1^\tau \alpha_n$.

For any $X \subseteq \mathbb{N}$, we write $(U_n)_{n \in X}$ for “the sequence defined for all $i \in X$ by U_i ”, and if U is a finite sequence and V is an arbitrary sequence, we also write $U.V$ for the concatenation of the two sequences, i.e. the sequence $(W_n)_{n \geq 0}$ such that for all $i \in \mathbb{N}$, $i < |U| \Rightarrow W_i = U_i$, and $i \geq |U| \Rightarrow W_i = V_{i-|U|}$. Moreover, when it is clear from the context, we will write $U.x$ where x is an element, here considered as a sequence of a single element.

A sequence of $k \in \mathbb{N} \cup \{\infty\}$ assemblies $\alpha_0, \alpha_1, \dots$ over \mathcal{A}^T is a \mathcal{T} -*assembly sequence* if, for all $1 \leq i < k$, $\alpha_{i-1} \rightarrow_1^\tau \alpha_i$. Alternatively, a \mathcal{T} -assembly sequence can be treated as a sequence of attachments $(t_i@z_i)_{i < k}$. The notation “ $t_i@z_i$ ” means “a tile of type t_i attaches to a position z_i ”.

The set of *productions* of a tile assembly system $\mathcal{T} = (T, \sigma, \tau)$, written $\mathcal{A}[\mathcal{T}]$, is the set of all assemblies producible from σ . An assembly α is called *terminal* if there is no β such that $\alpha \rightarrow_1^\tau \beta$. The set of terminal assemblies is written $\mathcal{A}_\square[\mathcal{T}]$.

For $A, B \in \mathbb{Z}^2$, we use the notation \vec{AB} to mean “the vector from A to B ”.

2.2 Intrinsic simulations

These definitions are standard definitions of simulations in self-assembly. They are for a large part taken from [20], and adapted to the space graph formalism used in this paper.

Let T be a tile set, and let $m \in \mathbb{N}$. An m -block supertile over T is a partial function $\alpha : \mathbb{Z}_m^2 \dashrightarrow T$ (in the square grid), $\alpha : \mathbb{H}_m \dashrightarrow T$ (in the hexagonal grid), or $\alpha : \mathbb{Z}_m^3 \dashrightarrow T$ (in the three-dimensional cubic grid), where $\mathbb{Z}_m = \{0, 1, \dots, m-1\}$ and $\mathbb{H}_m = \{a\vec{i} + b\vec{j} + c\vec{k} \mid a, b, c \in \{0, \dots, m-1\}\}$.

Let B_m^T be the set of all m -block supertiles over T . The m -block with no domain is said to be *empty*. For a general assembly $\alpha : G \dashrightarrow T$ (where $G = \mathbb{Z}^2$ in the square grid, $G = \mathbb{H}$ in the hexagonal grid, and $G = \mathbb{Z}^3$ in the 3D cubic grid), and $\vec{v} \in G$, define $\alpha_{\vec{v}}^m$ to be the m -block supertile defined by $\alpha_{\vec{v}}^m(\vec{v}') = \alpha(m\vec{v} + \vec{v}')$ for all \vec{v}' in \mathbb{Z}_m^2 , \mathbb{H}_m and \mathbb{Z}_m^3 , respectively.

For some tile set S , a partial function $R : B_m^S \dashrightarrow T$ is said to be a *valid m -block supertile representation* from S to T if for any $\alpha, \beta \in B_m^S$ such that $\alpha \sqsubseteq \beta$ and $\alpha \in \text{dom } R$, then $R(\alpha) = R(\beta)$.

For a given valid m -block supertile representation function R from tile set S to tile set T , define the *assembly representation function*² $R^* : \mathcal{A}^S \rightarrow \mathcal{A}^T$ such that $R^*(\alpha') = \alpha$ if and only if $\alpha(A) = R(\alpha_A^m)$ for all $A \in G$. For an assembly $\alpha' \in \mathcal{A}^S$ such that $R(\alpha') = \alpha$, α' is said to map *cleanly* to $\alpha \in \mathcal{A}^T$ under R^* if for all non empty blocks α_A^m , $A + \vec{u} \in \text{dom } \alpha$ for some $\vec{u} \in G$ such that $\|\vec{u}\|_1 \leq 1$.

In other words, α' may have tiles on supertile blocks representing empty space in α , but only if that position is adjacent to a tile in α . We call such growth “around the edges” of α' *fuzz* and thus restrict it to be adjacent to only valid supertiles, but not diagonally adjacent (i.e. we do not permit *diagonal fuzz*).

By extension, the representation function from a cubic tileset S to a planar square tileset T is a regular 3D representation function where we consider the operations of tileset T to happen in the $z = 0$ plane.

In the following definitions, let $\mathcal{T} = (T, \sigma_T, \tau_T)$ be a tile assembly system, let $\mathcal{S} = (S, \sigma_S, \tau_S)$ be a tile assembly system, and let R be an m -block representation function $R : B_m^S \rightarrow T$.

Definition 2.1. We say that \mathcal{S} and \mathcal{T} have equivalent productions (under R), and we write $\mathcal{S} \Leftrightarrow \mathcal{T}$ if the following conditions hold:

1. $\{R^*(\alpha') \mid \alpha' \in \mathcal{A}[\mathcal{S}]\} = \mathcal{A}[\mathcal{T}]$.
2. $\{R^*(\alpha') \mid \alpha' \in \mathcal{A}_{\square}[\mathcal{S}]\} = \mathcal{A}_{\square}[\mathcal{T}]$.
3. For all $\alpha' \in \mathcal{A}[\mathcal{S}]$, α' maps cleanly to $R^*(\alpha')$.

Another important condition that we require for simulation, is the equivalence of dynamics, via the two following definitions. The first one requires that whenever the simulator is capable of assembly a new supertile of some type t , the resulting production represents the assembly of a new tile in the simulated system:

Definition 2.2. We say that \mathcal{T} follows \mathcal{S} (under R), and we write $\mathcal{T} \dashv_R \mathcal{S}$ if $\alpha' \rightarrow^S \beta'$, for some $\alpha', \beta' \in \mathcal{A}[\mathcal{S}]$, implies that $R^*(\alpha') \rightarrow^T R^*(\beta')$.

²Note that R^* is a total function since every assembly of S represents *some* assembly of T ; the functions R and α are partial to allow undefined points to represent empty space.

The other direction is not as easy to define: intuitively, we require that whenever the simulated system is capable to assemble a new tile of type t , the simulator can also assemble a supertile representing t . However, because supertiles do not always determine which tile they represent by a single tile placement, but sometimes by many tile additions, the definition needs more care.

However, our results do not use the fully detailed version of this definition, instead requiring only that if a production p of the simulated system is not terminal, then no production of the simulator representing p is terminal.

Definition 2.3. *We say that \mathcal{S} models \mathcal{T} (under R), and we write $\mathcal{S} \models_R \mathcal{T}$, if for every $\alpha \in \mathcal{A}[\mathcal{T}]$, there exists $\Pi \subset \mathcal{A}[\mathcal{S}]$ where $R^*(\alpha') = \alpha$ for all $\alpha' \in \Pi$, such that, for every $\beta \in \mathcal{A}[\mathcal{T}]$ where $\alpha \rightarrow^{\mathcal{T}} \beta$, (1) for every $\alpha' \in \Pi$ there exists $\beta' \in \mathcal{A}[\mathcal{S}]$ where $R^*(\beta') = \beta$ and $\alpha' \rightarrow^{\mathcal{S}} \beta'$, and (2) for every $\alpha'' \in \mathcal{A}[\mathcal{S}]$ where $\alpha'' \rightarrow^{\mathcal{S}} \beta'$, $\beta' \in \mathcal{A}[\mathcal{S}]$, $R^*(\alpha'') = \alpha$, and $R^*(\beta') = \beta$, there exists $\alpha' \in \Pi$ such that $\alpha' \rightarrow^{\mathcal{S}} \alpha''$.*

The previous definition essentially specifies that every time \mathcal{S} simulates an assembly $\alpha \in \mathcal{A}[\mathcal{T}]$, there must be at least one valid growth path in \mathcal{S} for each of the possible next steps that \mathcal{T} could make from α which results in an assembly in \mathcal{S} that maps to that next step.

Definition 2.4. *We say that \mathcal{S} simulates \mathcal{T} (under R) if $\mathcal{S} \Leftrightarrow_R \mathcal{T}$ (equivalent productions), $\mathcal{T} \dashv_R \mathcal{S}$ and $\mathcal{S} \models_R \mathcal{T}$ (equivalent dynamics).*

2.3 Intrinsic Universality

Now that we have a formal definition of what it means for one tile system to simulate another, we can proceed to formally define the concept of intrinsic universality, i.e., when there is one general-purpose tile set that can be appropriately programmed to simulate any other tile system from a specified class of tile systems.

Let REPR denote the set of all supertile representation functions (i.e., m -block supertile representation functions for all $m \in \mathbb{Z}^+$). Define \mathfrak{C} to be a class of tile assembly systems, and let U be a tileset.

Definition 2.5. *We say U is intrinsically universal for \mathfrak{C} at temperature $\tau' \in \mathbb{Z}^+$ if there are functions $\mathcal{R} : \mathfrak{C} \rightarrow \text{REPR}$ and $S : \mathfrak{C} \rightarrow \mathcal{A}_{<\infty}^U$ such that, for each $\mathcal{T} = (T, \sigma, \tau) \in \mathfrak{C}$, there is a constant $m \in \mathbb{N}$ such that, letting $R = \mathcal{R}(\mathcal{T})$, $\sigma_{\mathcal{T}} = S(\mathcal{T})$, and $\mathcal{U}_{\mathcal{T}} = (U, \sigma_{\mathcal{T}}, \tau')$, $\mathcal{U}_{\mathcal{T}}$ simulates \mathcal{T} at scale m and using supertile representation function R .*

That is, $\mathcal{R}(\mathcal{T})$ is a representation function that interprets assemblies of $\mathcal{U}_{\mathcal{T}}$ as assemblies of \mathcal{T} , and $S(\mathcal{T})$ is the seed assembly used to program tiles from U to represent the seed assembly of \mathcal{T} .

Definition 2.6. *We say that U is intrinsically universal for \mathfrak{C} if it is intrinsically universal for \mathfrak{C} at some temperature $\tau' \in \mathbb{Z}^+$.*

3 Simulations and the Bisimilarity Lemma

3.1 The Bisimilarity Lemma

In order to look at local phenomena in the assembly, we define *restricted dynamics*, where we are only concerned about what happens within a region of the space. This is the intuition behind the *policy set* of a tile assembly system. Moreover, its *diplomatic set* is the set of its possible communications with the rest of the space, through its border.

Definition 3.1 (Glue movies). *Let $\mathcal{T} = (T, \sigma, \tau)$ be a tile assembly system on some graph $G = (V, E)$.*

Let $F \subseteq E$ be a set of edges of G . A glue movie on F is a sequence m of glue additions, where for all $i \geq 0$, a glue addition m_i is defined by $m_i = (e_i, s_i, g_i) \in F \times \{+, -\} \times \text{Glues}(T)$. We call e_i the edge of the glue addition, s_i its orientation (i.e. on which side of the edge is the new tile attached), and g_i its type. Moreover, no two glue additions in a movie can have the same edge e_i ; formally, for all $i, j \geq 0$ such that $i \neq j$, $e_i \neq e_j$.

A glue movie is therefore a sequence of glue additions, oriented in a canonical way: for any $F \subseteq E$, an assembly sequence induces a (possibly empty) glue movie along the edges of F , in the following way: whenever a new tile t attaches to the assembly using edges of F , we add to the glue movie all the edges of F where t matches the existing assembly, starting in clockwise order from the North of t . Moreover, the orientation $o \in \{+, -\}$ that we choose is $+$ for the south and west glues of t , and $-$ for its north and east glues (for the glues that are added to the glue movie).

Definition 3.2 (Policy and diplomatic sets). *Let $\mathcal{T} = (T, \sigma, \tau)$ be a tile assembly system on some graph $G = (V, E)$. Moreover, let $W \subseteq V$ be a set of vertices of G with a local origin, and $F \subseteq E$ be the set of edges between W and $V \setminus W$, with the same local origin. The policy set $\mathcal{P}(\mathcal{T}, W)$ of \mathcal{T} is the set of the restrictions to vertices of W of all possible assembly sequences of \mathcal{T} (including all sequences not corresponding to a terminal assembly). The diplomatic set $\mathcal{D}(\mathcal{T}, F)$ of \mathcal{T} along F is the set of glue movies on F defined by all possible assembly sequences of all productions of \mathcal{T} .*

Definition 3.3. *Let $\mathcal{T} = (T, \sigma, \tau)$ be a tile assembly system on some graph $G = (V, E)$. Let $X \subseteq E$ be a cut-set separating G into two connected components Y and Z such that $\text{dom } \sigma \subseteq Y$.*

Let $P = \mathcal{P}(\mathcal{T}, Z)$, $D = \mathcal{D}(\mathcal{T}, X)$, and let P' be a set of prefixes of elements of P . The restriction of D to P' , noted $D|_{P'}$, is the set of all prefixes of D where glues are only put next to tiles which are attached in P' .

Lemma 3.4 (Bisimilarity Lemma). *Let $\mathcal{T} = (T, \sigma, \tau)$ be a tile assembly system on some graph $G = (V, E)$. There is a map f , from diplomatic sets to policy sets, such that for any cut-set $X \subseteq E$ separating G into two connected components Y and Z such that $\text{dom } \sigma \subseteq Y$, $\mathcal{P}(\mathcal{T}, Z) = f(\mathcal{D}(\mathcal{T}, X))$.*

(In other words, the policy set of a tile assembly system, in a zone Z not containing any tile of σ , depends only on the diplomatic set on the border of Z).

Moreover, f is continuous: for any such X, Y, Z , take P' to be a set of prefixes of $\mathcal{P}(\mathcal{T}, Z)$, P' can be obtained as $f(D|_{P'})$.

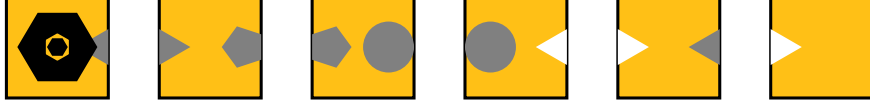


Figure 2: A simplistic tile assembly system

Note that for any X, Y , $\mathcal{D}(\mathcal{T}, X) = \mathcal{D}(\mathcal{T}, Y)$ implies that Y is obtained by a translation of X , as otherwise $\mathcal{D}(\mathcal{T}, X)$ and $\mathcal{D}(\mathcal{T}, Y)$ are not even defined on the same set.

3.2 Use of Lemma 3.4

Before we prove the Bisimilarity Lemma, let us review some of its uses and, for the reader familiar with the Window Movie Lemma [20], how it differs from that lemma.

Let us first consider the system S defined on Figure 2. It is quite clear that its productions are all $1 \times n$ rectangles, with $n \bmod 4 = 1$. We want to prove that it has a final production whose shape is a $1 \times n$ rectangle for arbitrarily large n . This can be proved “by observation”, by exhibiting the relevant production for infinitely many values of n . But giving a proof through the Bisimilarity Lemma (Lemma 3.4) will provide insight into how to apply the lemma to cases where observation of the tileset is infeasible.

First, let us notice that no production S has a tile outside of the $x = 0$ line. Let us consider for all $i > 0$ the zone $Z_i = \{(x, y) | x > i, y \in \mathbb{Z}\}$. For any i , $\mathcal{D}(S, Z_i)$ is a set of glue movies on the set of edges $\partial Z_i = \{((i, y), (i+1, y)) | y \in \mathbb{Z}\}$. These glue movies have at most two glue additions, at $\{(i, 0), (i+1, 0)\}^+$ and $\{(i, 0), (i+1, 0)\}^-$. There are 6 different glues in S , therefore there are at most $36 + 6 + 1 = 43$ possible glue movies of length 0, 1 or 2^3 . Thus, there are at most 2^{43} different values of $\mathcal{D}(S, Z_i)$.

Since S has a production of length greater than 2^{43} (this can be seen by actually building such a production), there are $i < j$ such that $\mathcal{D}(S, Z_i) = \mathcal{D}(S, Z_j)$, and $\mathcal{D}(S, Z_i)$ is not reduced to the empty glue movie. By the Bisimilarity Lemma, $\mathcal{P}(S, Z_j) = \mathcal{P}(S, Z_i)$. Because of this, for any production p of S , there is a production p' such that $p' \cup Z_j$ is $(p \cup Z_i)$ translated by $(j-i, 0)$. But then, p' is $j-i$ tiles longer than p , from which we get the result.

This proof shows how to “pump” an assembly using the Bisimilarity Lemma. In this example, the same result can be achieved by the Window Movie Lemma.

The proof of Theorem 1.1 shows how the two lemmas differ.

3.3 Proof of Lemma 3.4

Proof. We first define f for singletons: let S be a glue movie on X . We build a set $f(\{S\}) \subseteq (Z \times T)^{\mathbb{N}}$ of assembly sequences of \mathcal{T} , by induction on the length of S .

³Actually, the glue movie is determined by the first glue, so there are only 7 possible movies.

First let $f_0 = \{\emptyset\}$. Then, for all $i \geq 0$, let $(e_i, s_i, g_i) = S_i$ be tile attachment of index i in the assembly sequence.

- If $e_i = (a, b)$, $b \in Z$ (respectively $a \in Z$) and $s_i = +$ (respectively $s_i = -$), let T_i be the set of assembly sequences not involving any glue of X , that can grow from some assembly induced by an assembly sequence of f_i , and that is compatible with all the glue additions of S involving vertex b (respectively vertex a).

Remark that there are at most four such glue additions in S , and that, by the construction of an induced movie, they are all consecutive in S . Also, by this definition, any prefix of an assembly sequence of T_i is also in T_i .

Then, let $S_{i+1} = S_i \cup T_i$.

- Else, let $S_{i+1} = S_i$.

If S is finite, of length n , this construction stops after $m \leq n$ steps, and we let $f(\{S\}) = S_m$. Else, this construction never stops. Let $f_\infty = \bigcup_{i \in \mathbb{N}} f_i$, and $f(\{S\})$ be the union of f_∞ and the set of all limits of increasing sequences (with respect to the prefix order) of words from f_∞ . Moreover, for any diplomatic set D , $f(D)$ is defined by $f(D) = \bigcup_{m \in D} f(\{m\})$.

Finally, we prove that our construction for f satisfies our claims:

f is complete, i.e. $\mathcal{P}(\mathcal{T}, Z) \subseteq f(\mathcal{D}(\mathcal{T}, X))$. Let α be a finite production of $\mathcal{A}[\mathcal{T}]$, and S_α be an assembly sequence of α . By an immediate induction on the length of S_α , its restriction to Z is in $f(\mathcal{D}(\mathcal{T}, X))$.

Now, if α is an infinite production of $\mathcal{A}[\mathcal{T}]$, any assembly sequence S_α of α is also in $f(\mathcal{D}(\mathcal{T}, X))$: indeed, let M_α its induced glue movie on Z . Either M_α uses the edges of X only a finite number of times, and therefore one of its suffixes is included in T_i for some i , or it uses the edges of X an infinite number of times, and then the construction of $f(\{S_\alpha\})$ has infinitely many steps, each with a finite number of tile additions.

Therefore, in this case, all finite prefixes of S_α are in $f(\mathcal{D}(\mathcal{T}, X))$. Since S_α itself is the limit of these prefixes, it is therefore also in $f(\mathcal{D}(\mathcal{T}, X))$.

f is sound, i.e. $f(\mathcal{D}(\mathcal{T}, X)) \subseteq \mathcal{P}(\mathcal{T}, Z)$. We now prove that for any assembly sequence $s \in f(\mathcal{D}(\mathcal{T}, X))$, there is a valid assembly sequence s^β of some $\beta \in \mathcal{A}[\mathcal{T}]$ such that s is the restriction of s^β to Z .

Since $s \in f(\mathcal{D}(\mathcal{T}, X))$, there is a glue movie $m \in \mathcal{D}(\mathcal{T}, X)$ on X such that $s \in f(\{m\})$. By definition of the diplomatic set, m is the induced movie of some assembly sequence r of \mathcal{T} .

Now, we build s^β by induction: we will build a sequence $(s_n^\beta)_{n \in \mathbb{N}}$ of assembly sequences, placing at least one tile at each step, with the invariants that for all $n \geq 0$, s_n^β is a valid assembly sequence of \mathcal{T} , and the restriction of s_n^β to Z is a prefix of s .

We need a few auxiliary sequences, also defined inductively: $(a_n)_{n \in \mathbb{N}}$ is the sequence of indices in r , starting with a_0 , defined as the smallest integer such that r_{a_0} is placed in Y , and $(u_n)_{n \in \mathbb{N}}$ is the sequence of indices in s , starting at $u_0 = 0$.

- If possible, let $b_n \geq a_n$ be the largest integer such that all tile attachments of $r_{a_n, a_n+1, \dots, b_n}$ be in Y , and $s_n = s_n^\beta \cdot r_{a_n, a_n+1, \dots, b_n}$ be a valid assembly sequence.

In this case, let $s_{n+1}^\beta = s_n$, and let $a_{n+1} > b_n$ be the smallest integer such that $r_{a_{n+1}}$ is a tile attachment in Y . Also, let $u_{n+1} = u_n$.

- else, if movie m is infinite, or if $i < |m|$, we can move forward in m : indeed, the next step of r (at $b_n + 1$) necessarily involves a glue addition on some edge of X . Moreover, since m is the movie from which we built s , tile attachment s_{u_n} can be done in β^n . Let thus $s_{n+1}^\beta = s_n^\beta \cdot s_{u_{n+1}}$, and let $u_{n+1} = u_n + 1$. Also, let $a_{n+1} = a_n$.

□

3.4 Differences with known results

Remark how this definition and lemma differs from the window movie lemma [20]: by Lemma 3.4, if two diplomatic sets are the same up to translation, we can “swap” their whole policy sets (formally, the same tile assembly system can also produce an assembly with the part between the windows taken out). In other words, we can move partial assemblies back and forth between these two translations of the window, at *any* step of the dynamics.

In contrast, the window movie lemma uses a weaker hypothesis: in our formalism, it just asks for two diplomatic sets to intersect. It also yields a weaker conclusion, by showing that we can “swap” translations of partial assemblies *only once*.

4 Systems without mismatches do not simulate general systems

This section proves Theorem 1.1; the proof is split into three subsections: Section 4.1 shows that we can switch back and forth between the cutspace and \mathbb{Z}^2 , without changing geometric properties of the simulation. Section 4.2 shows a slightly stronger statement, namely that the dynamics are also the same in \mathbb{Z}^2 and in the cutspace. Altogether, these section intuitively show that the tileset cannot “detect” that it is being ran in the cutspace. Finally, Section 4.3 shows how to reconcile sequences that happen in the cutspace into a valid sequence in \mathbb{Z}^2 , that a claimed simulator without mismatches must be able to grow.

We say that two adjacent tiles *do not match* if at least one of them has a strictly positive glue on their common side, and their respective colors on that side are different.

Theorem 1.1. *There is a tile assembly system $\mathcal{T} = (T, \sigma, 1)$, such that for all simulator \mathcal{U} of \mathcal{T} , there are assemblies $\alpha \in \mathcal{A}[\mathcal{U}]$ where at least two adjacent tiles of α do not match on their abutting side.*

Proof. Let $\mathcal{T} = (T, \sigma, 1)$ be the tile assembly system with the tiles of Figure 3, and σ is the bottom-left tile of Figure 3, at position $(0, 0)$.

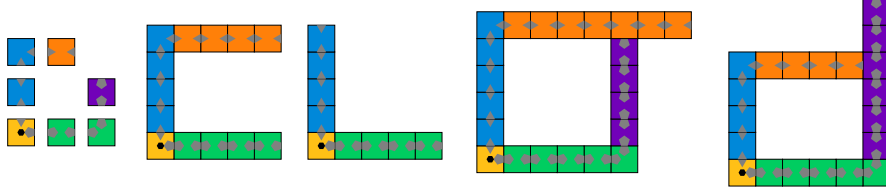


Figure 3: A tile assembly system at temperature 1; the seed tile is marked with a hexagon; in the pictured productions, the *upper arm* is in dark, and the *lower arm* in light.

One can readily see that the productions of \mathcal{T} are all one of the shapes shown on Figure 3. More precisely, they each have one *upper* and one *lower* arm (as pictured on Figure 3); if the bottom arm turns left and grows upwards, and the upper arm turns right and grows to the right, then the two arms compete at the intersection. The first to arrive continues towards infinity, and the other “crashes” into it.

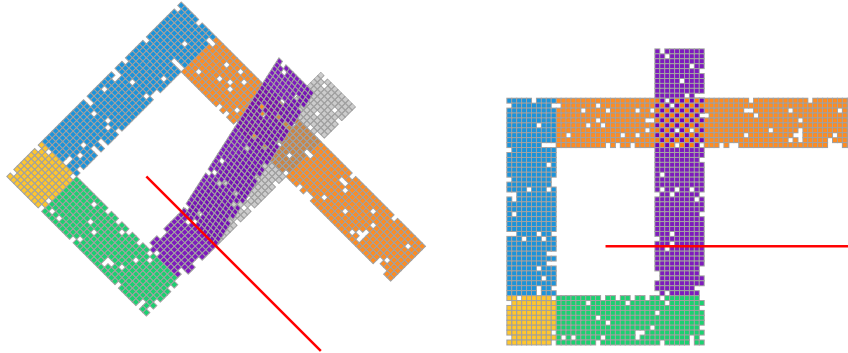


Figure 4: Left: Conflict between the two arms is averted in the cutspace: the upper arm passes “over” the lower arm. Right: trying to embed this production in \mathbb{Z}^2 might lead to a conflict in the intersection zone unless the tiles there coincide.

Now, assume for the sake of contradiction, that there is a tile assembly system \mathcal{U} , that simulates \mathcal{T} with scale factor s , and that \mathcal{U} does so without mismatches. We will show that \mathcal{U} is also able to assemble a production where the two arms cross, and both grow infinitely far. Since this production does not represent any production of $\mathcal{A}[\mathcal{T}]$, this will contradict our hypothesis that \mathcal{U} can simulate \mathcal{T} .

4.1 The cut-space, and how to embed assemblies of \mathbb{Z}^2 into it, that preserve diplomatic and policy sets

First, let us define an auxiliary assembly grid $G = (V, E)$, called the *cut-space*, where the set V of vertices is made of two copies of \mathbb{Z}^2 : $V = V_0 \cup V_1$, where $V_0 = \{(x, y, 0) | (x, y) \in \mathbb{Z}^2\}$ and $V_1 = \{(x, y, 1) | (x, y) \in \mathbb{Z}^2\}$. Let E_0 and E_1 be the set of edges within each copy of \mathbb{Z}^2 , and for $i \in \{0, 1\}$, let $E_i = \{((x, y, i), (x', y', i)) | x, y, x', y' \in V \text{ and } |x - x'| + |y - y'| = 1\}$.

The set E of edges of the cut-space is defined by exchanging the ends of the edges along a ray defined in \mathbb{R}^2 by $y = 2s + 0.5$ and $x \geq 2$. More precisely:

$$\begin{aligned} E &= (E_1 \cup E_0) \\ &\quad \setminus \{((x, 2s, i), (x, 2s + 1, i)) | x \geq 2, i \in \{0, 1\}\} \\ &\quad \cup \{((x, 2s, i), (x, 2s + 1, 1 - i)) | x \geq 2, i \in \{0, 1\}\} \end{aligned}$$

The *projection* π of the cutspace G into \mathbb{Z}^2 is the map defined on G by $\pi(x, y, z) = (x, y)$.

All positions in the cutspace still have four neighbors. Remark that the geometry of this graph makes the intuitive model of assemblies with square tiles less intuitive, as two surfaces could be assembled, that cross each other. However, in our construction, we will only use tiles in one direction at these positions, the left arm growing upwards. The idea of defining this space is to grow both assemblies of Figure 3 in the same space, as shown on Figure 4.

For each $y \geq 3$, the set $C_y = \{((x, y, 1), (x, y + 1, 1)) | x \in \mathbb{Z}\}$ is a cut-set of the cutspace. Also, for each $x \geq 3$, $D_x = \{((x, y, 0), (x, y + 1, 0)) | y > 3\} \cup \{((x', y, 0), (x' + 1, y, 0)) | x' \geq x\}$ is a cut-set of the cutspace. Moreover, $E_k = C_k \cup D_k$ is also a cut-set of G , separating it into three parts, shown on Figure 5. Likewise, define $A_{>y} = \{(x, y', 1) | x \in \mathbb{Z}, y' > y\}$, $B_{>x} = \{(x', y, 0) | y > 3 \wedge x' > x\}$ and $V_{>k} = A_{>k} \cup B_{>k}$ to be the “far side” of C_x , D_y and E_k . We will use these parts to describe the growth of the arms “past the crossing”.

Now, let W, H be two integers. Let p be a production of \mathcal{T} in \mathbb{Z}^2 in which the lower arm has run W tiles to the right and one tile upwards, and the upper arm has run H tiles upwards and one tile to the right. We know that continuing to run \mathcal{T} from p is going to enclose a $W \times H$ rectangle.

Since \mathcal{U} simulates \mathcal{T} with scale factor s , there is a production q of \mathcal{U} which is a representation of p . From now on, we will study productions which can be obtained from q . Define an auxiliary tile assembly system $\mathcal{U}' = (U, q, \tau)$, which is \mathcal{U} with q as the new seed. Remark that any production of \mathcal{U}' is also producible by \mathcal{U} .

We do not know what is the behaviour of \mathcal{U}' in the cutspace; it does not follow from the definition of simulation that it should behave like \mathcal{T} does in the cutspace.

Let ϵ be an embedding, mapping positions $z = (x, y)$ with $y \geq 2$ and $x \geq 2$ to $\epsilon(z) = (x, y, 1)$, and the other positions to $\epsilon(z) = (x, y, 0)$. ϵ maps adjacent positions in \mathbb{Z}^2 to adjacent positions in the cutspace.

Let $u = (t_i @ z_i)_{i \in \mathbb{N}}$ be an assembly sequence of \mathcal{U}' in \mathbb{Z}^2 . Define $\epsilon(u)$ in the cutspace as the embedding of this sequence, i.e. the sequence defined by $(t_0 @ \epsilon(z_0)), (t_1 @ \epsilon(z_1)), \dots, (t_n @ \epsilon(z_n))$. Observe that $\epsilon(u)$ is an assembly sequence of \mathcal{U}' in the cutspace, since adjacency is preserved under ϵ . Its upper arm lives in the 0 layer of the cutspace, and its lower arm in the 0 layer for its horizontal

part, and in the 1 layer for its upwards part. Therefore, any assembly sequence of \mathcal{U}' in \mathbb{Z}^2 yields by ϵ an assembly sequence of \mathcal{U}' in the cut-space where the arms are at least as long.

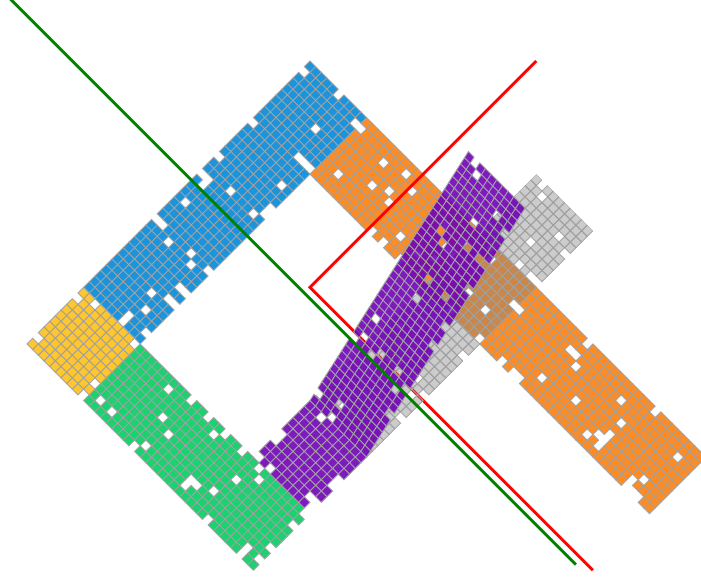


Figure 5: E_k partitions the cut-space into three parts. On this figure, C_k is in green, and D_k is in red. Note that C_k does *not* intersect with the blue part of the assembly (upper left), but floats over it. Likewise, D_k does not intersect the purple arm.

Conversely, take an assembly sequence v of \mathcal{U}' in the cutspace which produces p where either the upper arm has length less than sH , or the lower arm has length less than sW . Then for all $z, z' \in p$ such that $z \neq z'$, $\pi(z) \neq \pi(z')$. Therefore, by mapping π over v , one gets a configuration $\pi(p)$ which is a production of \mathcal{U}' on \mathbb{Z}^2 . Thus, if an assembly sequence of \mathcal{U}' in the cutspace places a tile further away than s tiles from the arms, it does so after both arms have passed position (sH, sW, l) , where l is their respective layer. On the other hand, if H and W are large enough, namely, if $N = \min(H, W) > 2^{(4|U|+1)^{6s} \cdot (6s)!}$, then there must be $k, k' < N$ such that the prefixes of the diplomatic sets on E_k and E'_k before any tile can be put further away than s from the arms are the same, since all prefixes of movies before that event have at most a finite number of tile addition. By the laziness property of Lemma 3.4, the prefix of the policy sets of $V_{>k}$ and $V_{>k'}$ before any tile placement outside of the arms must be the same. But then, a minimal assembly sequence which adds a tile outside the arms must have each arm have length less than k' , which leads to a contradiction. Therefore, \mathcal{U}' only places tiles within s tiles of the position of the arms on the cutspace, just as it does on \mathbb{Z}^2 .

Conclusion of this section: for large enough W and H , there are two

indices k, k' such that $0 < k < k' < \min(W, H)$ such that the full diplomatic sets $\mathcal{D}(\mathcal{U}', E_k)$ and $\mathcal{D}(\mathcal{U}', E_{k'})$ (in the cut space) are the same, up to a translation by a vector $(k' - k, 0, 0)$ for D_k , and by a vector $(0, k' - k, 0)$ for C_k (which is possible because D_k and C_k are disjoint). Moreover, if $W > 4Ns$ and $H > 4Ns$, k and k' can be taken such that $k' - k > 3s$.

4.2 Getting the arms to cross in the cut space

The first lemma we will need is essential to prove that we can “keep the dynamics going”, i.e. that we can add tiles, in the cut space, until the two arms “cross” (of course, since they are not on the same plane, the bottom arm “jumps over” the top arm).

One issue is, although we are simulating a system in \mathbb{Z}^2 which, if ran in the cut space, would produce two infinite arms nicely crossing each other, the simulation of this system in \mathbb{Z}^2 could send signals between the arms to detect that it is being ran in the cut space, and stop the growth of one of the arms, allowing a successful simulation.

However, the following lemma shows that no tiling can use such a strategy. To show this, we use once again the bisimilarity lemma:

Lemma 4.1. *Let u be an assembly sequence of \mathcal{U}' in the cutspace, which produces some assembly p . If p does not have at least one tile at $(s(H+1)+1, y_H, 0)$ (i.e. strictly to the right of the bottom arm) for some y_H (respectively, $(x_W, s(W+1)+1, 1)$ for some x_W , i.e. strictly above the top arm), then p is not terminal. Moreover, in any attachment sequence u' having u as a prefix, there is an integer $i > |u|$ such that u'_i is an attachment on the lower arm (respectively, the upper arm).*

Proof. Let u be an assembly sequence ending in a production p which does not have a tile at $(s(H+1)+1, y_H, 0)$ nor at $(x_W, s(W+1)+1, 1)$ for any y_H and x_W . By the Bisimilarity Lemma applied to $E_{k'}$ and E_k , one can get a corresponding assembly sequence u' which does not have a tile at $(s(H-2)+1, y_H, 0)$ nor at $(x_W, s(W-2)+1, 1)$ for any y_H and x_W (see Figure 6). But $\pi(u')$ is a valid assembly sequence in \mathbb{Z}^2 , and in any of its prolongations, both arms must get at least s rows or columns past (Hs, Ws) .

Therefore, in the policy set of $E_{>k}$, there must be a way to extend u further than $s(H-1)$ to the right on layer 0 or without tiles further upwards than $s(W-1)$ on layer 1. Thus, by translating by $(k-k')$ to get the policy set of $E_{>k'}$, we get the expected result. \square

Note that this lemma states a property relating *all* assembly sequences of \mathcal{U}' in two translated regions. Hence, its proof needs the full Bisimilarity Lemma (except the continuity of f): indeed, the Window Movie Lemma would allow us to work only on *one* assembly sequence.

4.3 Unifying assembly sequences of crossing arms, in the projection of the cut-space onto \mathbb{Z}^2

Definition 4.2. *Let u be an sequence of attachments of \mathcal{U}' in the cutspace. u is consistent if for any $t@z, t'@z' \in u$, if $\pi(z) = \pi(z')$, then $z = z'$*

For a consistent sequence of attachments u , we define $\Pi(u)$ to be the subsequence of first occurrences of elements in $(\pi(x))_{x \in u}$. In other words for each positions whose projection appears twice in u , Π only keeps the first occurrence.

Lemma 4.3. *Let u be a consistent assembly sequence of \mathcal{U}' in the cutspace, and let p be the production assembled by u . Let $i < |u'|$, and $t@_i(x, y, l) = u'_i$ such that $y > 2s + 1$. Then, if $(x, y, 1 - l)$ is attachable in p , $t@_i(x, y, 1 - l)$ is a valid attachment onto p , and thus, $u.t@_i(x, y, 1 - l)$ is a consistent assembly sequence.*

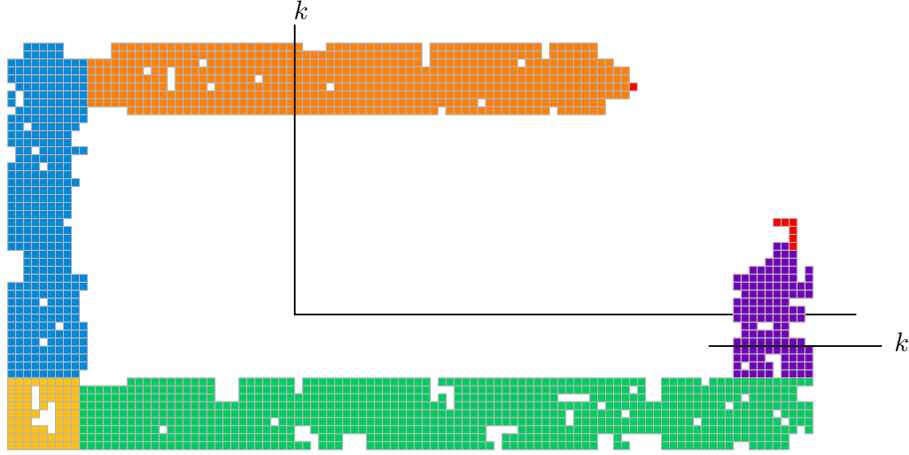


Figure 6: A production (in the cutspace) where both arms are short must be able to place at least one tile on each arm. These tiles are drawn in red.

Proof. Since u is a consistent sequence, $\Pi(u)$ is an assembly sequence of \mathcal{U}' in \mathbb{Z}^2 , which yields $\pi(p)$ as production. Since $(x, y, 1 - l)$ is attachable and $y > 2s + 1$, there are some neighbors of $(x, y, 1 - l)$ in p with total glue abutting $(x, y, 1 - l)$ at least τ , and they are on layer $1 - l$. These neighbors map by π onto neighbors of (x, y) , and since $\pi(p)$ is without mismatches, the glues they have on their sides abutting with (x, y) match those of t . Therefore, $t@_i(x, y, 1 - l)$ is a possible attachment in p . \square

We finish the proof by iteratively reconciling assembly sequences until the two arms cross in \mathbb{Z}^2 , yielding the desired contradiction:

Lemma 4.4. *There is an integer n_0 and a sequence $(u^n)_{n \leq n_0}$ of consistent assembly sequences in the cutspace such that for some y_h, x_v, l, l' , u^{n_0} has an attachment at $((s + 1)H + 1, y_h, l)$ and one at $(x_v, (s + 1)V + 1, l')$. Moreover, for $1 \leq n \leq n_0$, $|u^n| > |u^{n-1}|$.*

With u^n given by Lemma 4.4, $\Pi(u^{n_0})$ is an assembly sequence of \mathcal{U}' in \mathbb{Z}^2 with the two arms crossing. Thus, \mathcal{U}' has a production which does not represent any production of \mathcal{T} . From this, it follows that \mathcal{U} does not simulate \mathcal{T} .

Proof. Let u^0 be an empty sequence. For all $n \geq 0$, u^{n+1} is defined as follows:

Success. If $\Pi(u^n)$ is a consistent assembly sequence of \mathcal{U}' with the two arms crossing (i.e., with tiles at positions $((s+1)H+1, y_H, l)$ and $(x_V, (s+1)V+1, l')$ for some y_H and x_V), then $n_0 = n$.

Non-conflicting case. Else, u^n does not represent a terminal assembly because of Lemma 4.1. Therefore, a new attachment $t@z$ is possible after u^n in the arm that is too short. If $u^n.t@z$ is a consistent sequence, then let $u^{n+1} = u^n.t@z$.

Conflicting case. Lastly, if $u^n.t@z$ is not consistent, then by Lemma 4.3, there is a t' such that $u^n.t'@z$ is consistent. In that case, let $u^{n+1} = u^n.t'@z$.

Condition “Success” must be reached after a finite number of steps, since only a finite number of attachments can be made in each arm before a tile is placed at both $((s+1)H+1, y_H, l)$ and $(x_V, (s+1)V+1, l')$. This proves that the desired sequence does exist and concludes the proof.

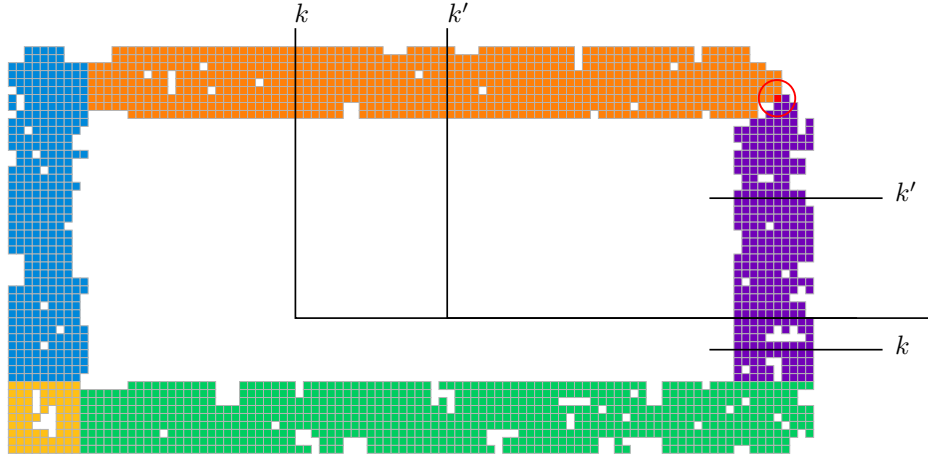


Figure 7: The first conflict in the projection of the cut-space into \mathbb{Z}^2 is on the tile circled in red.

□

□

5 Local consistency is strictly stronger than the mismatch-free property

Local consistency is the property of a tile assembly system that no production has mismatches, and that all tiles are always placed such that there is no *excess of glue*, i.e. such that the sum of their glue strengths is *exactly* equal to the temperature. This property has been used in [11] to construct an intrinsically universal tileset for a restricted class.

However, the classical mismatch-free constructions of Turing machine simulations [27, 31] are also locally consistent. Moreover, the recent separation result of temperatures 1 and 2 [20], also applies to separate locally consistent

temperature 2 systems from temperature 1. This raises the question of whether local consistency is stronger than the simpler mismatch-free property, which we answer with the following theorem:

Theorem 1.2. *There is a mismatch-free tile assembly system $\mathcal{T} = (T, \sigma, 1)$, such that no tile assembly system \mathcal{U} (of any temperature) simulating \mathcal{T} is locally consistent.*

Proof. The idea is similar to the proof of Theorem 1.1: we will exhibit a very simple system that does not have mismatches but is not locally consistent, and then repeatedly use Lemma 3.4 on a claimed simulator \mathcal{U} of \mathcal{T} to show that at least one tile, in a production of \mathcal{U} , must be placed with too much glue.

In this proof, \mathcal{T} will be the temperature 1 system with one uniform tile, one copy of which is its seed tile. \mathcal{T} is described on Figure 8.

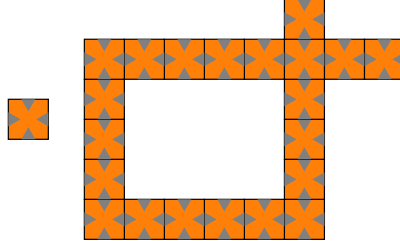


Figure 8: Tile assembly system \mathcal{T} , with an example production. All glues have strength 1. Remark that the tiles at the junctions of two branches can always be placed with an excess of glue.

We will show that any simulator \mathcal{U} for \mathcal{T} , at some scaling factor f , has either a mismatch or an excess of glue, for at least one tile attachment. First, \mathcal{T} can produce an assembly α with two parallel arms, a bottom arm and a top arm, both of length at least N , where N is the constant defined as in the proof of Theorem 1.1, and where the bottom arm turns left after N steps (i.e. a prefix of the construction shown on Figure 8).

By hypothesis, \mathcal{U} must be able to produce an assembly β representing α at scaling factor f . Let β_0 be the largest prefix of β such that the two arms do not cross, and let γ_0 be the assembly equivalent to β_0 in the cut-space defined in the proof of Theorem 1.1. By Lemma 3.4, there are two integers k and k' , with $k' \geq k + 10f$ such that diplomatic sets $\mathcal{D}(\mathcal{U}, E_k)$ and $\mathcal{D}(\mathcal{U}, E_{k'})$ are translations of each other. We can therefore get an assembly sequence δ in the cutspace where both arms are “long”, i.e. pass position $f.N$ in their respective layer, by the same argument as in Lemma 4.1 (namely, because we can find an assembly with “short” versions of the arms, that can place at least one more tile because such an assembly cannot represent a terminal assembly).

However, the projection of δ by π hits some position twice, since the bottom and top arms would cross in \mathbb{Z}^2 . Let thus s^0 be the longest prefix of s yielding an assembly δ^0 whose projection by π hits each position in \mathbb{Z}^2 at most once. We claim that at least one tile placed by s^0 either has a mismatch with one of its neighbors, or else is placed with an excess of glue. Indeed, the tile it places

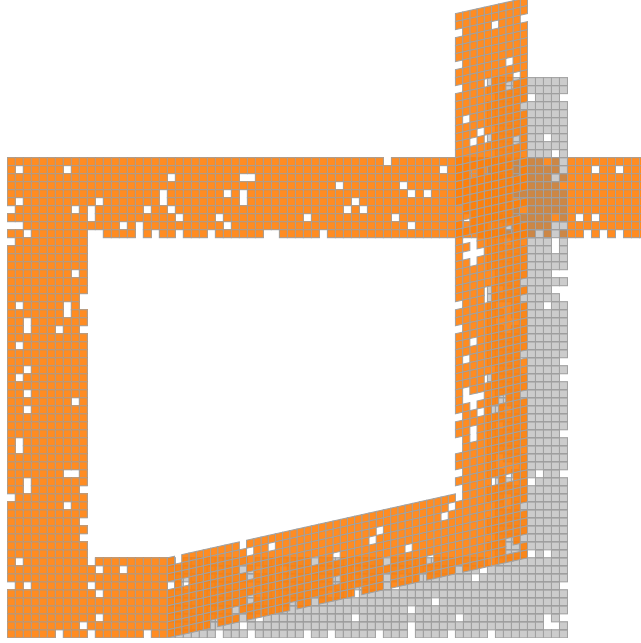


Figure 9: Assembly δ (in the cutspace)

at step $|s^0|$ must be overlapping some previously placed tile s_i . Let j be the last index strictly before $|s^0|$ such that s_j is a tile placement adjacent to $s_{|s^0|}$. This index exists because there must be at least one tile adjacent to $s_{|s^0|}$ before placing it.

Moreover, s_j must be in the same layer of the cut-space as $s_{|s^0|}$, because $s_{|s^0|}$ is not adjacent to an edge between the two layers. Therefore, s_j is not in the same layer as s_i , and so these two tiles can be placed independently. Let now s^1 be the smallest prefix $s_{0,1,\dots,\max(i,j)}$ of s containing both tiles. If $\pi(s_{\max(i,j)}^1)$ matches all its neighbors, then it can be placed with an excess of glue: indeed, the sum of the glue strengths with its neighbors in the cut-space is at least τ , and it has a non-zero glue strength in common with $\pi(s_{\min(i,j)}^1)$ (since $s_{\max(i,j)}^1$ interacts with $s_{|s^0|}$).

□

6 3d Systems without mismatches simulate themselves

We saw that systems without mismatches do not have the same power as general aTAM systems. We now investigate whether they have an intrinsically universal system, that is, whether there is a tileset on a given grid which can simulate all systems without mismatches on that same grid without itself making mismatches. In this section, we will show that in \mathbb{Z}^3 , there is such a universal tileset for system without mismatches.

Theorem 1.3. *There is a three dimensional tileset U such that for every 3d tile assembly system T without mismatches on \mathbb{Z}^3 , there is a seed $s_U(T)$ such that tile assembly system $(U, s_U(T), 2)$ (defined on \mathbb{Z}^3) simulates T without making mismatches.*

6.1 Simulating 2d with 3d

The full construction for Theorem 1.3 is somehow hard to visualize, as three-dimensional constructions tend to be. We will first prove a restricted version, in which the *simulated* system is two-dimensional. The gist of both proofs is essentially the same.

Theorem 6.1. *There is a 3D tileset U without mismatches that is intrinsically universal at temperature 2 for the class of 2D mismatch-free tile assembly systems.*

Proof. In order to prove this, we rely on standard tools such as lookup tables and Turing machine simulations, as described in other constructions of intrinsically universal tilesets [10, 11].

One particular difficulty is to not create mismatches, which are used in crucial parts of the cited constructions. Although this is relatively straightforward when simulating *deterministic* systems without mismatches, the main issue is to deal with non-determinism: indeed, if two different tiles can be placed at the same position by different cooperation, as shown on Figure 10, the cut-space technique and the bisimilarity lemma could be used, like in the proof of Theorem 1.1, to reconcile inconsistent encodings of the north glue, and produce a “cancerous” glue.

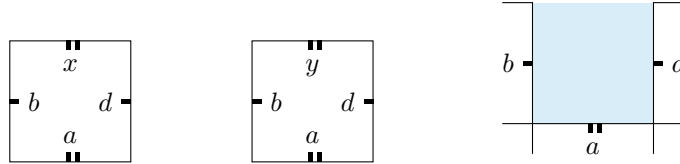


Figure 10: Both tiles on the right-hand side can attach to the empty location in blue on the assembly on the left-hand side, in different ways: for instance, b and d could cooperate and place the first tile, outputting an x on the north side, or a could place the second tile, outputting a y . The problem, if we stay in \mathbb{Z}^2 , is that techniques such as those used in the proof of Theorem 1.1 seem to allow us to “mix” these two sequences in one, outputting an inconsistent encoding of a mix of x and y on the north side.

The main idea of this construction is to allow different assembly sequences to take place at the same time, but at different positions. The key to doing this is to embed the construction into \mathbb{Z}^3 . In our construction, we use four layers: one to compute output sides, using different copies of the computational gadget for all possible input side combinations. One to choose (non-deterministically) the particular combination that will be used, preventing others to interfere with that choice, and two other layers to propagate information between all sides.

6.1.1 Construction of a macrotile, step 1: gathering information

The global layout of a macrotile is pictured on Figure 12: the lowest layer is the leftmost one, the highest is the rightmost one. The construction of a macrotile *starts at one of the intermediate layers*, when a neighboring tile places the encoding of (k, g, T) on one of these two layers, where k is the size of the gadgets (from which the scale factor is calculated), g is the glue on the neighboring tile, and T is the simulated tileset.

We take as an example the assembly of the first tile of Figure 10, from the east: glue d comes first, as an encoding of (k, d, T) on the second layer (the *H-wireboard* layer). This encoding sends a “probe” containing only an encoding of k and T , to the bottom layer. Then, a one-way wire is built, that goes around the bottom layer. This wire actually uses a counter to count up to the scale factor and turn at the appropriate places, until reaching the south side. The choice of the south as the preferred side is actually independent of where this probe started: all decisions are taken on the south side.

Since this counter does not depend on the types of glues that might come later around this supertile, all these glues will send the same probe; we can therefore design the probes, using appropriate counters, so that different probes grown concurrently have no mismatches when this happens.

When it reaches the south side, this probe non-deterministically chooses $\log |T|$ bits. It then sends two different pieces of information to the two intermediate layers: (1) an “unblocking” wire consisting of only one tile, that instructs all neighboring glues to send information to the highest layer (where lookups are performed), and (2) the $\log |T|$ bits chosen.

Using these two pieces of information, all the gadgets in the highest layer that deal with east glues start to compute outputs: thanks to the $\log |T|$ chosen bits, this choice can be done uniformly among all gadgets.

Then, *if another input arrives*, say, from the west, it will start the same probe to the bottom layer, where the probe and the already grown path agree, by our construction. In parallel to that, since the unblocking signal has already been sent by the bottom layer, the wires on the intermediate layers can start to grow, sending information to the gadgets on the highest layer, that can process the west side. These gadgets can finally start to compute potential outputs, using the encoding of T and the $\log |T|$ “non-deterministic” bits. As detailed in the next section, all these gadgets compute different subsets of the input sides: therefore, only gadgets expecting only the east side will be able to start their computation, and, if T can actually place a tile at the current position, will output the corresponding glues.

Clearly, the two intermediate layers can transmit all this information, both by having “holes” (to let the $\log |T|$ “non-deterministic bits” go through the layers), and by forming “wires” to connect glues to the highest layer, according to the routing scheme pictured by the full lines of Figure 11, and detailed later in the next section.

6.1.2 Construction of a macrotile, step 2: computing the output

The main purposes of the two intermediate layers, called *H-wireboard* and *V-wireboard* on Figure 12 is to duplicate the information of the neighboring glues and of the $\log |T|$ non-deterministic bits chosen on the bottom layer, and to

bring them to the gadgets on the top layer.

We describe these gadgets in greater detail now: they are actually made of standard lookup tables, as used for instance in [10]; these constructions are described in the literature [31, 36] and do not use mismatches. We only need to arrange them properly in the space, so that neighboring gadgets do not interfere with each other. More precisely, we use the following routing scheme: if the supertile in construction has a supertile on its east side, that supertile sends two copies of the required information (T , the value of the glue, and the scale factor), drawn in purple on Figure 11, to the west. Similarly, when the supertile in construction has a tile on its west side, that supertile tile sends two copies of the same information to the east, but at different positions, shown in green on Figure 11. A similar scheme is used to route glue information from the north and south sides (shown in blue and orange on Figure 11).

This information is then routed on their respective rows and columns, until it reaches gadgets that can handle it. On Figure 11, we can see that all combinations of input sides are indeed handled by different gadgets.

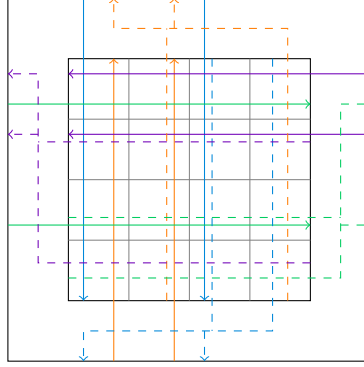


Figure 11: Routing scheme: the outer rectangle is the macrotile, the inner one is the gadget area, divided into 16 gadgets. The inputs are routed by the full lines, and the outputs by the dashed line. On this figure, the two layers of routing (horizontal and vertical) are shown together. In the real construction, vertical and horizontal wires are exclusively in the V-wireboard and in the H-wireboard, respectively.

6.1.3 Construction of a macrotile, step 3: outputting glues

After computation takes place in the gadgets, it is time to construct glues on the remaining sides. To do so, we use the dashed lines shown on Figure 11. The information is completely output in each of the gadgets, routed by these lines to a “peripheral” area around the gadgets area, duplicated into two copies, and sent to the correct positions to start the next supertile.

The only issue that needs to be handled here, is the case where two different gadgets are activated, and send their result to the same wire (one of the wires in dashed line on Figure 11). Since the $\log |T|$ bits chosen at stage 1 fixed the type of the supertile that will be placed (among the set of tiles that can attach there), the values sent on these wires are always equal: we can therefore simply

use “wires” that can grow in both directions on these output rows, and remain consistent with other values that might be output by other gadgets.

Finally, because we are simulating a mismatch-free system, the glues output on abutting sides of two neighboring tiles are always equal, so we can design their encoding so that there is no mismatch at these positions. \square

6.2 Simulating 3d with 3d

We are now ready to give the proof of Theorem 1.3.

Theorem 1.3. *There is a three dimensional tileset U such that for every 3d tile assembly system T without mismatches on \mathbb{Z}^3 , there is a seed $s_U(T)$ such that tile assembly system $(U, s_U(T), 2)$ (defined on \mathbb{Z}^3) simulates T without making mismatches.*

Proof. Essentially, the construction is the same, it just needs four times more instances of the gadget on the same plane, and, as a consequence, some changes in the wiring to accept input from the nadir and zenith. A scheme of this construction is shown on Figure 13.

To route the input from the nadir and zenith to the wireboards, we add one more instance of the routing scheme shown on Figure 11. These new wires need some space to reach the bottom of the gadgets: for that, we use six wires (three inputs, three outputs) on each side, instead of four in the previous construction. \square

7 Acknowledgements

The authors wish to thank Damien Woods for friendly and invaluable discussions, ideas, comments, improvements and support before and while writing this paper.

References

- [1] Pablo Arrighi, Nicolas Schabanel, and Guillaume Theyssier. Intrinsic simulations between stochastic cellular automata. Technical report, 2012. arXiv:1208.2763 [cs.FL].
- [2] Robert Berger. *Undecidability of the Domino problem*. Memoirs of the American Mathematical Society, 1965.
- [3] W. P. Blackstock and M. P. Weir. Proteomics: quantitative and physical mapping of cellular proteins. *Trends Biotechnol.*, 17(3):121–127, Mar 1999.
- [4] Eric Goles Ch., Pierre-Étienne Meunier, Ivan Rapaport, and Guillaume Theyssier. Communication complexity and intrinsic universality in cellular automata. *Theoretical Computer Science*, 412(1-2):2–21, 2011.
- [5] Matthew Cook, Yunhui Fu, and Robert T. Schweller. Temperature 1 self-assembly: deterministic assembly in 3D and probabilistic assembly in 2D. In *Proceedings of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 570–589, 2011. Arxiv preprint: arXiv:0912.0027.

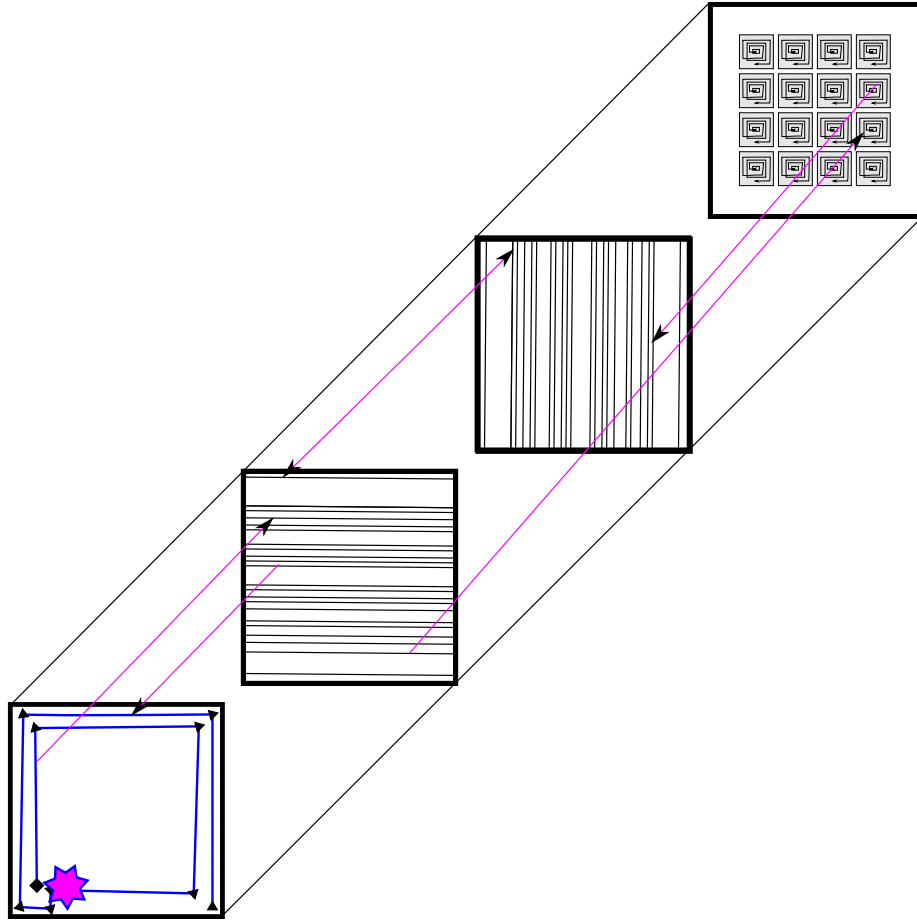


Figure 12: Layout of a macro-tile of U simulating T . The top layer has 16 *loci*, one for each possible combination of input sides, the middle layers are the wireboards, with the *H-wireboard* carrying the E and W inputs and outputs between the edges and the loci, and the bottom layer, the *V-wireboard* carrying the N and S inputs / outputs. Finally, the bottom layer is responsible for making a non-deterministic choice for the macrotile and distributing it to the system wires on the wireboards. This choice takes place in the starred location, after all input sides have had a chance to trigger the choice layer. The magenta arrows represent the inter-layer communication.

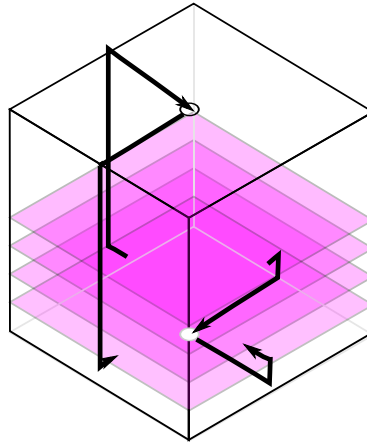


Figure 13: Layout of a macrotile; the shaded planes are the four planes corresponding to the previous simulation the wiring from the edges of these planes to the border of the macrotile is not represented, for clarity's sake.

- [6] Marianne Delorme, Jacques Mazoyer, Nicolas Ollinger, and Guillaume Theyssier. Bulking I: an abstract theory of bulking. *Theoretical Computer Science*, 412(30):3866–3880, 2011.
- [7] Marianne Delorme, Jacques Mazoyer, Nicolas Ollinger, and Guillaume Theyssier. Bulking II: Classifications of cellular automata. *Theor. Comput. Sci.*, 412(30):3881–3905, 2011.
- [8] Erik D. Demaine, Martin L. Demaine, Sándor P. Fekete, Matthew J. Patitz, Robert T. Schweller, Andrew Winslow, and Damien Woods. One tile to rule them all: Simulating any tile assembly system with a single universal tile. In Javier Esparza, Pierre Fraigniaud, Thore Husfeldt, and Elias Koutsoupias, editors, *ICALP (1)*, volume 8572 of *Lecture Notes in Computer Science*, pages 368–379. Springer, 2014.
- [9] Erik D. Demaine, Matthew J. Patitz, Trent A. Rogers, Robert T. Schweller, Scott M. Summers, and Damien Woods. The two-handed tile assembly model is not intrinsically universal. In *ICALP: 40th International Colloquium on Automata, Languages and Programming*, volume 7965 of *LNCS*, pages 400–412, Riga, Latvia, July 2013. Springer. Arxiv preprint: [arXiv:1306.6710](#).
- [10] David Doty, Jack H. Lutz, Matthew J. Patitz, Robert T. Schweller, Scott M. Summers, and Damien Woods. The tile assembly model is intrinsically universal. In *Proceedings of the 53rd Annual IEEE Symposium on Foundations of Computer Science*, pages 439–446, October 2012. Arxiv preprint: [arXiv:1111.3097](#).
- [11] David Doty, Jack H. Lutz, Matthew J. Patitz, Scott M. Summers, and Damien Woods. Intrinsic universality in self-assembly. In *Proceedings of the 27th International Symposium on Theoretical Aspects of Computer Science*, pages 275–286, 2009. Arxiv preprint: [arXiv:1001.0208](#).

- [12] David Doty, Matthew J. Patitz, and Scott M. Summers. Limitations of self-assembly at temperature 1. *Theoretical Computer Science*, 412(1–2):145–158, 2011. Arxiv preprint: [arXiv:0906.3251](#).
- [13] Constantine Glen Evans. *Crystals that count! Physical principles and experimental investigations of DNA tile self-assembly*. PhD thesis, California Institute of Technology, 2014.
- [14] Kenichi Fujibayashi, Rizal Hariadi, Sung Ha Park, Erik Winfree, and Satoshi Murata. Toward reliable algorithmic self-assembly of DNA tiles: A fixed-width cellular automaton pattern. *Nano Letters*, 8(7):1791–1797, 2007.
- [15] Grégory Lafitte and Michael Weiss. Universal tilings. In Wolfgang Thomas and Pascal Weil, editors, *STACS 2007, 24th Annual Symposium on Theoretical Aspects of Computer Science, Aachen, Germany, February 22–24, 2007, Proceedings*, volume 4393 of *Lecture Notes in Computer Science*, pages 367–380. Springer, 2007.
- [16] Grégory Lafitte and Michael Weiss. Simulations between tilings. In *Conference on Computability in Europe (CiE 2008), local proceedings*, pages 264–273, 2008.
- [17] Grégory Lafitte and Michael Weiss. An almost totally universal tile set. In Jianer Chen and S. Barry Cooper, editors, *Theory and Applications of Models of Computation, 6th Annual Conference, TAMC 2009, Changsha, China, May 18–22, 2009. Proceedings*, volume 5532 of *Lecture Notes in Computer Science*, pages 271–280. Springer, 2009.
- [18] Kyle Lund, Anthony T. Manzo, Nadine Dabby, Nicole Micholotti, Alexander Johnson-Buck, Jeanetter Nangreave, Steven Taylor, Renjun Pei, Milan N. Stojanovic, Nils G. Walter, Erik Winfree, and Hao Yan. Molecular robots guided by prescriptive landscapes. *Nature*, 465:206–210, 2010.
- [19] Ján Maňuch, Ladislav Stacho, and Christine Stoll. Two lower bounds for self-assemblies at temperature 1. *Journal of Computational Biology*, 17(6):841–852, 2010.
- [20] Pierre-Étienne Meunier, Matthew J. Patitz, Scott M. Summers, Guillaume Theyssier, Andrew Winslow, and Damien Woods. Intrinsic universality in tile self-assembly requires cooperation. In *Proceedings of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 752–771, 2014. Arxiv preprint: [arXiv:1304.1679](#).
- [21] Nicolas Ollinger. Universalities in cellular automata a (short) survey. In *JAC*, pages 102–118, 2008.
- [22] Matthew J. Patitz, Robert T. Schweller, and Scott M. Summers. Exact shapes and Turing universality at temperature 1 with a single negative glue. In *DNA 17: Proceedings of the Seventeenth International Conference on DNA Computing and Molecular Programming*, LNCS, pages 175–189. Springer, September 2011. Arxiv preprint: [arXiv:1105.1215](#).

- [23] Lulu Qian and Erik Winfree. Scaling up digital circuit computation with DNA strand displacement cascades. *Science*, 332(6034):1196, 2011.
- [24] Lulu Qian, Erik Winfree, and Jehoshua Bruck. Neural network computation with DNA strand displacement cascades. *Nature*, 475(7356):368–372, 2011.
- [25] R.M. Robinson. Undecidability and nonperiodicity for tilings of the plane. *Inventiones Mathematicae*, 12:177–209, 1971.
- [26] Paul W. K. Rothemund. Folding DNA to create nanoscale shapes and patterns. *Nature*, 440(7082):297–302, March 2006.
- [27] Paul W. K. Rothemund and Erik Winfree. The program-size complexity of self-assembled squares (extended abstract). In *STOC '00: Proceedings of the thirty-second annual ACM Symposium on Theory of Computing*, pages 459–468, Portland, Oregon, United States, 2000. ACM.
- [28] Paul W.K. Rothemund, Nick Papadakis, and Erik Winfree. Algorithmic self-assembly of DNA Sierpinski triangles. *PLoS Biology*, 2(12):2041–2053, 2004.
- [29] Georg Seelig, David Soloveichik, David Yu Zhang, and Erik Winfree. Enzyme-free nucleic acid logic circuits. *Science*, 314(5805):1585–1588, 2006.
- [30] Nadrian C. Seeman. Nucleic-acid junctions and lattices. *Journal of Theoretical Biology*, 99:237–247, 1982.
- [31] David Soloveichik and Erik Winfree. Complexity of compact proofreading for self-assembled patterns. In *The eleventh International Meeting on DNA Computing*, 2005.
- [32] Hao Wang. Proving theorems by pattern recognition – II. *The Bell System Technical Journal*, XL(1):1–41, 1961.
- [33] Hao Wang. Dominoes and the AEA case of the decision problem. In *Proceedings of the Symposium on Mathematical Theory of Automata (New York, 1962)*, pages 23–55. Polytechnic Press of Polytechnic Inst. of Brooklyn, Brooklyn, N.Y., 1963.
- [34] Bryan Wei, Mingjie Dai, and Peng Yin. Complex shapes self-assembled from single-stranded DNA tiles. *Nature*, 485(7400):623–626, 2012.
- [35] Erik Winfree. *Algorithmic Self-Assembly of DNA*. PhD thesis, California Institute of Technology, June 1998.
- [36] Erik Winfree. Simulations of computing by self-assembly. Technical Report CaltechCSTR:1998.22, California Institute of Technology, 1998.
- [37] Erik Winfree, Furong Liu, Lisa A. Wenzler, and Nadrian C. Seeman. Design and self-assembly of two-dimensional DNA crystals. *Nature*, 394(6693):539–44, 1998.
- [38] Damien Woods. Intrinsic universality and the computational power of self-assembly. 2013. Arxiv preprint: [arXiv:1309.1265](https://arxiv.org/abs/1309.1265).

- [39] Bernard Yurke, Andrew J Turberfield, Allen P Mills, Friedrich C Simmel, and Jennifer L Neumann. A DNA-fuelled molecular machine made of DNA. *Nature*, 406(6796):605–608, 2000.